Example for PL/I

This section contains examples of using direct Adabas calls in PL/ I . The previously defined Adabas files are used in each example.

```
/*** CONTROL BLOCK ***/
DCL 1    CONTROL_BLOCK,
    02   FILLER1               CHAR (2) INIT (’ ’),
    02   COMMAND_CODE          CHAR (2) INIT (’ ’),
    02   COMMAND_ID            CHAR (4) INIT (’ ’),
    02   FILE_NUMBER           BIN FIXED (15) INIT (0),
    02   RESPONSE_CODE         BIN FIXED (15) INIT (0),
    02   ISN                   BIN FIXED (31) INIT (0),
    02   ISN_LOWER_LIMIT       BIN FIXED (31) INIT (0),
    02   ISN_QUANTITY          BIN FIXED (31) INIT (0),
    02   FORMAT_BUFFER_LENGTH  BIN FIXED (15) INIT (100),
    02   RECORD_BUFFER_LENGTH  BIN FIXED (15) INIT (250),
    02   SEARCH_BUFFER_LENGTH  BIN FIXED (15) INIT (50),
    02   VALUE_BUFFER_LENGTH   BIN FIXED (15) INIT (100),
    02   ISN_BUFFER_LENGTH     BIN FIXED (15) INIT (20),
    02   COMMAND_OPTION_1      CHAR(1) INIT (’ ’),
    02   COMMAND_OPTION_2      CHAR(1) INIT (’ ’),
    02   ADDITIONS_1           CHAR(8) INIT (’ ’),
    02   ADDITIONS_2           CHAR(4) INIT (’ ’),
    02   ADDITIONS_3           CHAR(8) INIT (’ ’),
    02   ADDITIONS_4           CHAR(8) INIT (’ ’),
    02   ADDITIONS_5           CHAR(8) INIT (’ ’),
    02   COMMAND_TIME          BIN FIXED (31) INIT (0),
    02   USER_AREA             CHAR(4) INIT (’ ’);

/*** USER BUFFER AREAS ***/
DCL FORMAT_BUFFER    CHAR(100),
    RECORD_BUFFER    CHAR(250),
    SEARCH_BUFFER    CHAR(50),
    VALUE_BUFFER     CHAR(100),
    ISN_BUFFER       CHAR(20);
*
*

/***    ADDITIONAL FIELDS USED IN THE EXAMPLES ***/
DCL
    COMM_ID_X  BIN FIXED(31);
    COMM_ID    CHAR(4) BASED (ADDR(COMM_ID_X));
DCL      INPUT_KEY CHAR(8);
DCL      SYNC_CHECK_SWITCH CHAR(1) INIT(’0’);
DCL 1 RECORD_BUFFER_EX2,
         2   RECORD_BUFFER_A  CHAR(8),
         2   RECORD_BUFFER_B  DEC FIXED(3,0),
         2   FILLER3 CHAR(240);
DCL 1 RECORD_BUFFER_EX3,
         2   OPEN_RECORD_BUFFER,
             3   OPEN_RECORD_BUFFER_X CHAR(8),
             3   FILLER4 BIN FIXED(31),
         2   FILLER5 CHAR(18),
         2   UPDATED_XC CHAR(6),
         2   LAST_XD CHAR(8),
         2   FILLER6 CHAR(5),
    1 USER_DATA,
         2   RESTART_XD CHAR(8),
         2   RESTART_ISN BIN FIXED(31);
DCL      ADABAS ENTRY OPTIONS(ASM);
```

Example 1

- Find the set of records in file 2 with XB = 99.
- Read each record selected using the GET NEXT option.

Issue Open Command

```
*** Issue Open Command **/
EXMP1:
    COMMAND_CODE = ’OP’;
    RECORD_BUFFER = ’ACC.’;
    CALL ADABAS (CONTROL_BLOCK,FORMAT_BUFFER,RECORD_BUFFER);
    IF RESPONSE_CODE > 0        THEN GOTO EX1ERR;
```

Issue Find Command

```
/*** Issue Find Command ***/
    COMMAND_CODE = ’S1’;
    COMMAND_ID = ’S101’;
    FILE_NUMBER = 2;
    ISN_LOWER_LIMIT = 0;
    ISN_BUFFER_LENGTH = 0;
    FORMAT_BUFFER = ’.’;
    SEARCH_BUFFER = ’XB,3,U.’;
    VALUE_BUFFER = ’099’;
    CALL ADABAS (CONTROL_BLOCK, FORMAT_BUFFER,
         RECORD_BUFFER, SEARCH_BUFFER, VALUE_BUFFER);
    IF RESPONSE_CODE > 0 THEN GOTO EX1ERR;
EX1A:
    IF ISN_QUANTITY = 0 THEN GOTO EX1EXIT;
EX1B:
    COMMAND_CODE = ’L1’;
    ISN = 0;
    COMMAND_OPTION_1 = ’N’;
    FORMAT_BUFFER  = ’RG.’;
EX1C:
    CALL ADABAS (CONTROL_BLOCK,FORMAT_BUFFER,RECORD_BUFFER);
    IF RESPONSE_CODE = 0 THEN
         GOTO EX1D;
    IF RESPONSE_CODE = 3 THEN
         GOTO EX1EXIT;
EX1D:
    . . .PROCESS RECORD . . .
         GOTO EX1C;
```

Error Routine

```
/*** Error Routine ***/
EX1ERR:
/*   . DISPLAY ERROR MESSAGE */
/*   . TERMINATE USER PROGRAM */
```

Issue Close Command

```
/** Issue Close Command **/
EX1EXIT:
    COMMAND_CODE = ’CL’;
    CALL ADABAS (CONTROL_BLOCK);
    IF RESPONSE_CODE > 0 THEN
         GOTO EX1ERR;
```

Example 2

- All records in file 1 are to be read in physical sequential order.
- Each record read is to be updated with the following values:
  - Field AA = ABCDEFGH
  - Field AB = 500
- User is to have exclusive control of file 1.

Issue Open Command

```
/*** Issue Open Command ***/
EXMP2:
    COMMAND_CODE = ’OP’;
    RECORD_BUFFER = ’EXU=1.’;
    CALL ADABAS (CONTROL_BLOCK,FORMAT_BUFFER,RECORD_BUFFER);
    IF RESPONSE_CODE > 0 THEN   GOTO EX2ERR;
```

Issue Read Physical Sequence Command

```
/*** Issue Read Physical Seq.  Command ***/
EX2A:
    COMMAND_ID = ’L201’;
    FILE_NUMBER = 1;
    ISN = 0;
    FORMAT_BUFFER = ’GA.’;
EX2B:
    COMMAND_CODE = ’L2’;
    CALL ADABAS (CONTROL_BLOCK,FORMAT_BUFFER,RECORD_BUFFER);
    IF RESPONSE_CODE = 0 THEN   GOTO EX2C;
    IF RESPONSE_CODE = 3 THEN   GOTO EX2EXIT;
    GOTO EX2ERR;
```

Update Record

```
/*** Update record.  ***/
/* Same fields are to be updated as were read.  */
/* Same CID and FORMAT BUFFER can be used for update.  */
/* ISN of record to be updated is already in ISN field as a result of */
/* the L2 command.  */
EX2C:
    COMMAND_CODE = ’A1’;
    RECORD_BUFFER_A = ’ABCDEFGH’;
    RECORD_BUFFER_B = 500;
    CALL ADABAS (CONTROL_BLOCK,FORMAT_BUFFER,
                 RECORD_BUFFER_EX2);
    IF RESPONSE_CODE > 0 THEN   GOTO EX2ERR;
    GOTO EX2B;
```

Error Routine

```
/*** Error Routine ***/
EX2ERR:
/*   . DISPLAY ERROR MESSAGE */
/*   . TERMINATE USER PROGRAM */
```

Close User Session

```
/* Close User Session */
EX2EXIT:
    COMMAND_CODE = ’CL’;
    CALL ADABAS (CONTROL_BLOCK);
    IF RESPONSE_CODE > 0 THEN   GOTO EX2ERR;
```

Example 3

This example illustrates a user session with ET logic. The user program is to perform the following functions:

1. During user session initialization, display information indicating the last successfully processed transaction of the previous user session.

2. For each user transaction:

- Accept from a terminal 8 characters of input that is used as the key for updating files 1 and 2.
- Issue a Find command for file 1 to determine if a record exists with field AA = input key.
- If no record is found, issue a message.
- If a record is found:
    - Delete the record from file 1;
    - Add a new record to file 2: Field RA = input key entered. Other fields to contain null value.
    - If the record cannot be successfully added, issue a BT command, display error message.
    - If both updates are successful, issue an ET command.

Session Initialization

This section of the program is only executed during user session initialization.

- The OP command is issued with ET data of the previous session being read.
- A message is displayed on the terminal screen identifying the last successfully processed transaction of the user’s previous session.

```
EX3:
    COMMAND_CODE = ’OP’;
    COMMAND_OPTION_2 = ’E’;
    ADDITIONS_1 = ’USER0003’;
    ADDITIONS_3 = ’PASSWORD’;
    RECORD_BUFFER = ’UPD=1,2.’;
    CALL ADABAS (CONTROL_BLOCK,FORMAT_BUFFER,RECORD_BUFFER);
    IF RESPONSE_CODE = 9 THEN   GOTO EX3;
    IF RESPONSE_CODE > 0 THEN
         GOTO EX3ERR;
EX3A:
    COMM_ID = COMMAND_ID;
    IF COMM_ID_X = 0 THEN
         GOTO EX3B;
/*  Display ET data (contained in RECORD BUFFER) on screen to inform user of
    last successfully processed transaction of previous user session. */
         . . .DISPLAY ET DATA. . .
    GOTO EX3C;
EX3B:
/*                          */
/*** No ET data received.  */
/*   Display message that no transactions were successfully processed during
    the previous user session. */
         . . .DISPLAY MESSAGE . . .
/*                               */
/*** Transaction processing.  ***/
/* This section is executed for each user transaction.  */
EX3C:
         . . .ACCEPT INPUT FROM TERMINAL. . .
/*                                                        */
/* Issue Find command for file 1 to determine if rec exists with field AA
   equal to input key entered. */
EX3D:
    COMMAND_CODE = ’S4’;
    COMMAND_ID = ’ ’;
    FILE_NUMBER = 1;
    ISN_LOWER_LIMIT = 0;
    FORMAT_BUFFER = ’.’;
    SEARCH_BUFFER = ’AA.’;
    VALUE_BUFFER = INPUT_KEY;
    CALL ADABAS (CONTROL_BLOCK,FORMAT_BUFFER,RECORD_BUFFER,
         SEARCH_BUFFER,VALUE_BUFFER,ISN_BUFFER);
    IF RESPONSE_CODE = 0 THEN
         GOTO EX3E;
    GOTO EX3ERR;
EX3E:
    IF ISN_QUANTITY > 0 THEN
         GOTO EX3F;
/*                                                        */
/* No record found, issue message requesting correction.  */
         . . .ISSUE MESSAGE . . .
    GOTO EX3C;
/*                              */
/* Delete record from file 1.  */
/* ISN of record to be deleted is already in ISN field and in hold
status
   as a result of the S4 command. */
EX3F:
    COMMAND_CODE = ’E4’;
    CALL ADABAS (CONTROL_BLOCK);
    IF RESPONSE_CODE = 0 THEN
         GOTO EX3G;
    IF RESPONSE_CODE = 9 THEN
         GOTO EX3D;
    GOTO EX3ERR;
/***Add new record to file 2.  */
EX3G:
    COMMAND_CODE = ’N1’;
    FILE_NUMBER = 2;
    FORMAT_BUFFER = ’RA.’;
    RECORD_BUFFER = INPUT_KEY;
    CALL ADABAS (CONTROL_BLOCK,FORMAT_BUFFER,RECORD_BUFFER);
    IF RESPONSE_CODE = 0 THEN
         GOTO EX3I;
    IF RESPONSE_CODE = 9 THEN
         GOTO EX3D;
/*                                */
/*  Attempt to add new record not successful. Backout transaction and
notify
    user that error condition exists. */
    COMMAND_CODE = ’BT’;
    CALL ADABAS (CONTROL_BLOCK);
    IF RESPONSE_CODE = 0 THEN
         GOTO EX3H;
/*                            */
/*  Backout not successful.   */
     . . .ISSUE MESSAGE INDICATING BACKOUT NOT SUCCESSFUL . .
    GO TO EX3ERR.
/*                               */
EX3H:
/*** Backout successful.    ***/
/* Issue message indicating error condition detected while adding new
record.*/
         . . .ISSUE MESSAGE. . .
    GOTO EX3ERR;
/*                                     */
/*** Updates successfully executed.  ***/
/*   Issue ET command with ET data.    */
EX3I:
    COMMAND_CODE = ’ET’;
    COMMAND_OPTION_2 = ’E’;
    RECORD_BUFFER = INPUT_KEY;
    CALL ADABAS (CONTROL_BLOCK,FORMAT_BUFFER,RECORD_BUFFER);
    IF RESPONSE_CODE = 0 THEN
         GOTO EX3C;
    IF RESPONSE_CODE = 9 THEN
         GOTO EX3D;
/*                                    */
/***    Error Routine    ***/
EX3ERR:
/*  . DISPLAY ERROR MESSAGE */
/*  . TERMINATE USER PROGRAM */
         . . .
```